Creating natural capital maps using python scripts

These rough notes describe how to use the set of python scripts developed by Alison Smith and Martin Besnier at the Environmental Change Institute, University of Oxford, for creating a habitat base map suitable for natural capital mapping. The scripts are research tools and have not been rigorously tested or documented.

The tools are designed to work with OS Mastermap (OSMM) as the base map. They then merge in a habitat map. The code is set up to work with either:

1. The Phase 1 habitat and land use (HLU) map for Oxfordshire, provided under license from the Thames Valley Record Centre (TVERC).
2. Natural England’s Priority Habitat Inventory – a freely available dataset covering all of England, which is less accurate and less detailed than the Phase 1 habitat data.

It should be possible to use similar datasets by modifying the input parameters for the main merge script, and possibly modifying the pre- and post-processing steps to work with the habitat classifications in the alternative datasets (if they differ).

The most straightforward way to merge together two or more habitat datasets would be though a series of Intersect or Identity operations in ArcGIS. However, the habitat datasets we work with tend to have boundaries that do not exactly match OS Mastermap boundaries. Therefore this method, when performed at county scale, creates tens of thousands of tiny ‘slivers’ at polygon edges, which can cause problems when attempting to perform subsequent geoprocessing operations. However, it would be possible to use a simple Identity operation instead of the ‘Merge into Base Map’ script if you are mapping a small area.

The novel aspect of the ‘Merge into Base Map’ code (designed originally by Martin Besnier, a visiting researcher from the Université Paris Sud) is that it merges two polygon datasets (a base map and a set of new features), while avoiding the creation of an excessive number of slivers due to boundary mismatch. It does this through identifying polygons in the base map that need to be split to match the boundaries of the new features, while ignoring minor differences in the boundaries due to inaccurate mapping (which would generate slivers). The output is faithful to the base map boundaries as far as possible, though minor differences (just a few cm) may arise during one of the sliver elimination steps.

A sequence of scripts have been developed which:

1. pre-process the OSMM and habitat input maps (e.g. to remove duplicate and overlapping features);
2. merge the two polygon datasets together without generating too many slivers;
3. generate a unified ‘Interpreted habitat’ attribute field that uses a set of rules to identify the habitat based on information from both OSMM and the habitat map;
4. Add in additional data from the Rural Payments Agency’s CROME crop dataset, to distinguish between arable land and pasture (‘improved grassland).

These scripts, plus a ‘MyFunctions’ script containing frequently used custom-built functions, form the core of the habitat base map generation. Further scripts (not described in this document) have also been developed that merge in Agricultural Land Class, up to 18 different habitat and historic designation datasets, OS Mastermap Greenspace and OS Open Greenspace, and public access information, to create a more detailed natural capital base map.

# Instructions for running the code

1. All the scripts should be in the same folder as the MyFunctions.py script, which supplies useful common functions. For large datasets, it is fastest to run the code on your hard drive rather than on a network server.
2. Download the geodatabase version of OSMM topographic area from Edina Digimap or OS (assuming you have a license). Do not use a shapefile version, as field names get truncated and code does not work.
3. The code is set up to work for two main options:
   1. Phase 1 habitat data is available. Check the habitat data for ‘unidentified’ habitats, and either use BAP / Section 41 habitat data or other info to determine what they are, or delete them (the OSMM definition will be used instead). Check for incorrect spellings in the Phase1habitat field (duplicates with upper and lower case can be left in, as they should be resolved by the code).
   2. Phase 1 habitat data is not available – use RPA CROME data and Natural England Priority Habitat Inventory (PHI) data instead. Download RPA CROME and NE PHI, Wood Pasture and Parkland (WPP), and Open Mosaic Habitats on previously developed land (OMHD) datasets for your area from data.gov.uk.
4. Set up a geodatabase containing the OS Mastermap input file (called either ‘OSMM’ if already clipped to the area boundary, or ‘OSMM\_in’ if you want the code to clip it for you) plus the outline boundary of your area.
5. If using Phase 1 habitat data, copy it to the gdb and call it ‘HLU’ (habitat and land use) if already clipped to the boundary, or ‘HLU\_in’ if not clipped yet. If using PHI, WPP and OMHD you will probably find it easiest to use the script ‘Setup\_LAD\_gdbs.py’ to pre-process these files – see below.
6. The CROME dataset can be either in this geodatabase or a different one. You need to pre-process it by:
   1. joining it to the CROME land-use code lookup table, which provides the land use description and a simplified description (‘Simple’) for each CROME land use code. The table is provided as a spreadsheet; import this to a GIS table using the ArcGIS ExcelToTable function. Then right click on the CROME dataset and select ‘Join’; join it to the table based on the lucode field; and export to a new dataset. Before exporting the joined CROME dataset, go to ‘Geoprocessing’, ‘Environment settings’, ‘Field’ and make sure that the box labelled ‘Maintain fully qualified field names’ is unchecked. This means that when you export the joined file, the new fields will be called just ‘Land\_Use\_Description’ and ‘Simple’, as in the lookup table, rather than having the lookup table name added in front of the field names (e.g. ‘CROME\_lookup.Land\_Use\_Description’).
   2. dissolving on land-use code (lucode), ‘Land\_Use\_Description’ and ‘Simple’ (do not create multi-part features, i.e. leave the multi-part box un-ticked when dissolving).
7. For each stage of the process, open the script (in IDLE or PyCharm), edit it to set the workspace directory and all the other parameters (see below), and run the code. In PyCharm, right click on the script tab at the top and select ‘Run..script name’. Caution – if you just select ‘Run’ from the box in the top right, make sure the correct script is showing in the dropdown next to the box otherwise it will just run the last script that you ran. Check the outputs carefully before moving on to the next step.

## Using Phase 1 habitat data

1. Open the **Merge\_OSMM\_HLU\_ Preprocess.py** script:

* Set the workspace gdb path in the code (NB - it is faster to run this on your hard drive rather than a network server). The code currently expects a geodatabase called Merge\_OSMM\_HLU\_CR\_ALC.gdb.
* Depending on what steps you want to run, change the flags in the parameter section to True or False. If the files have not been clipped to the exact boundaries, set ‘clip\_to\_boundary’ to True, name the input files HLU\_in and OSMM\_in, add the boundary file to the gdb and set the name of the boundary feature class (e.g. “Oxfordshire”) as a parameter in the code. Otherwise the code expects the input files to be named HLU and OSMM (this can be changed in the parameter section).
* Run the script. This erases manmade features and water from HLU, removes overlaps and eliminates slivers, checks and repairs geometry; removes overlapping features (landforms and pylons) from OSMM; and removes unnecessary fields (enter the list of habitat fields to keep in the script: currently "POLYID", "PHASE1HAB", "S41HABITAT", "SITEREF", "COPYRIGHT", "VERSION").

1. **Merge\_into\_Base\_MapV5b.py:** open the script, set the merge\_type parameter to “OSMM\_HLU”, check the workspace and all the other parameters and run the code. This can take about 14 hours for merging HLU habitat data into OSMM for the whole of Oxfordshire.
2. **OSMM\_HLU\_Interpret.py**: set ‘region’ to ‘Oxon’. This adds the habitat interpretations, by combining OSMM and Phase 1 appropriately. Note: If OSMM is "undefined" this usually means the area is under development or scheduled for development. The ‘undefined\_or\_original’ flag allows the user to choose whether to map these areas as "undefined" or as the current / original habitat pre-development.
3. **Merge\_CROME**.py: set ‘region’ to ‘Oxon’. This determines whether agricultural land is arable or improved grassland. It is currently set to only join to the CROME data where a CROME polygon overlaps at least 30% of the habitat polygon.

## Using CROME and NE PHI data

1. All scripts are designed to run in a loop through Local Authority Districts (LADs), with one geodatabase for each district, as the script was designed for analysing all the LADs in the OxCam Arc. If you are using a different area you can simply place a single geodatabase in the input folder.
2. **Prep\_OSMM.py** – (optional) this copies multiple OSMM tiles from download folders (e.g. downloaded from Edina) into a single folder, merges into a single feature class in a geodatabase, and clips to LAD boundaries. If you have a small area that covers only one or two OSMM tiles you do not need to do this – simply merge the tiles manually in ArcMap and copy the merged file to your gdb, calling it ‘OSMM\_in’, and when running Merge\_into\_Base\_Map.py set ‘clip\_to\_boundary’ to True.
3. **Setup\_LAD\_gdbs.py**. Has several stages – set any that are not needed to ‘False’.
   1. Prepare PHI data. There are three separate datasets – the main PHI data, plus Wood Pasture and Parkland (WPP) and Open Mosaic Habitats on previously developed land (OMHD). For all three datasets, the script dissolves on habitat field (Main\_habit or PRIHABTXT), converts to single part, deletes polygons <10m2, and copies the habitat field to a new field called ‘PHI’, ‘WPP’ or ‘OMHD’. Then it unions all three datasets.
   2. Set up individual geodatabases for each LAD (it is easier to do this manually, as described above, if you have just one area and you are not using LADs – so set all these stages to False in the header area, and clip the PHI data manually) :
      1. Create new gdb
      2. Copy in the OSMM data for this LAD
      3. Create boundary feature
      4. Clip Arc-wide PHI data to the boundaries of the LAD, and copy in.
4. **OSMM\_HLU\_Interpret.py** – set ‘region’ to ‘Arc’, and ‘simplify\_OSMM’ to True but other stages to False. This will interpret a simplified habitat from the OSMM Make, Descriptive Group and Descriptive Term.
5. **Merge CROME\_PHI.py:** This is a three step process:
   1. run **Merge\_CROME\_PHI.py** with ‘step = 1’. This should set ‘interpret\_PHI’ to False but all other steps to True. It merges in the CROME data.
   2. run **Merge\_into\_Base\_MapV5.py** with merge-type set to ‘Arc\_CROME\_PHI’, to merge in the PHI polygons.
   3. **re-run Merge\_CROME\_PHI .py** with ‘step = 2’. This should set ‘interpret\_PHI’ to True and all other stages to False, to copy the correct habitat interpretation across from the PHI, WPP and OMHD fields to the Interpreted\_habitat field. WPP and OHMD are not very accurate, e.g. WPP maps whole parkland areas with a mix of habitats including grass, fields, woods, buildings and plantations, and often overlaps with other PHIs. So we let OSMM woodland, water, manmade and other PHI habitats take priority.
6. **Tidy up the dataset and the gdb.** To keep only the fields you need, right click on the dataset, go to ‘Properties’ then ‘Fields’, and un-tick all the fields you do not need, then export to a new dataset. The fields to KEEP are: TOID, Make, Descriptive\_Group, Descriptive\_Term, OSMM\_hab, Interpreted\_habitat (this is the main habitat field), CROME\_desc and CROME\_simple. You can also delete all the intermediate datasets from the gdb. Keep OSMM, PHI, boundary, CROME, OSMM\_CROME and OSMM\_CROME\_PHI, plus whatever you called the new dataset after exporting only the fields you need to keep as described above.